SAM-GA

A Collection of Programs for Survival Analysis Methods For Genetic Analysis

http://cceb.upenn.edu/~hli/prog.html

Author:
Hongzhe Li, Ph.D
Department of Biostatistics and Epidemiology
University of Pennsylvania School of Medicine
Philadelphia, PA 19104-6021
Email: hli@cceb.upenn.edu

Update Information
1st version: June 2000
2rd version: June 2004
3rd version: June 2005.

Contents

1 INTRODUCTION

This documentation describes a collection of C/C++ programs and Splus/R functions for survival analysis methods for genetic analysis (SAM-GA). These programs were developed as part of Dr. Hongzhe Li research project "Survival Models for Mapping Genes for Complex Diseases", supported by NIH grant ES-009911 (1998-2005). See Table 1 for a brief description of these programs.

Table 1: Programs included in SAM-GA

Name	Source	Summary	Reference
CoxGENE	C	fit Cox model wit major gene effects	Li et al., 1998
CoxGENENVI	C	fit Cox model with major gene and random familial effects	Li and Thompson, 1997
SibLINK	Splus/R	linkage analysis of sibship age of onset data using population disease rates	Li and Zhong, 2002
PCRCox	Splus/R	partial Cox regression	Li and Gui, 2004
TGDCox	Splus/R	threshold gradient descent method for fitting the Cox model	Gui and Li, 2005
LarsCox	Splus	use LARS to fit L1 penalized Cox model	Gui and Li, 2005

This documentation lists all these programs, including a brief introduction of the models, the program INPUT/OUTPUT files, the compilation of the C/C++ programs. All programs and documentation can be downloaded from the website,

http://cceb.upenn.edu/∼hli/prog.html.

After downloading and extracting, consult this documentation which provides instructions for installing the program, using the Makefile, and for input data format and interpretation of the output. This documentation can not provide details of the methods implemented; readers will need to refer to the cited papers if unfamiliar with the objectives and background of a main program.

Please note that this documentation can describe only the programs available at a given time. The set of programs is under continuing development. Readers should check Hongzhe Li's web page

http://cceb.upenn.edu/∼hli

for updates.

2 The CoxGENE Package

2.1 Linked to PANGAEA

The programs distributed here need to call the Gibbs libraries maintained by Dr. Elizabeth Thompson at the University of Washington. These libraries are available by anonymous ftp
ftp://ftp.u.washington.edu/pub/user-supported/pangaea/PANGAEA/.
Two libraries from Dr. Thompson's Gibbs libraries *libnghds.a,librans.a* and three header files *nghds.h, quant.h ,rans.h* are needed. They are the Nghds library that contains pedigree structure set-up and routines, and the Rans library that contains routines for random number generation, and also routines for timing test programs, and data-stamping output. These files are available in MAORGAN V2.0.1 which can be downloaded from Dr. Thompson's web site given above. For each source codes of the library, there is a Makefile which provides instruction on how to install the libraries.

For both CoxGENE and CoxGENEENVI packages, Makefiles are provided with the programs to show how to link my programs with Dr. Thompson's Gibbs library. These Makefiles were developed for Sun Unix machine (both SPARC-station and SOLARIS-station). They should also be portable on other UNIX system (such as DEC Alpha) with some modifications. Section 2.3 and section 3.2 give more detailed discussion of the Makefiles for CoxGENE and CoxGENEENVI. For each package, we provide a test data set and the output file. It is recommended that you run the programs using the test data and check the output against the outputs we provide before attempting to run the programs on your owe data.

2.2 Brief Description of CoxGENE Routines

Table 2 gives a brief description of CoxGENE Routines. The majority of the programs source code files contain a brief comment section stating their purpose.

Table 2: Brief Description of CoxGENE Routines

coxseg.c	main program when allele frequency is estimated
inputdata.c	input data
output.c	output data
gene.c	gene manipulation
gibbs.c	Gibbs sampling procedure
stdcox1.c	procedure for Cox regression model
maxfun.c	maximization functions

2.3 Makefile for CoxGENE

The "LIB" and "Headers" in Makefile need to be changed properly. This Makefile will do several tasks like installing all libraries, linting all programs and libraries, and loading all main programs.

```
Makefile for CoxGENE
-------------------------------------------------------------------------
GIBBS = /people/biostat5/hongzhe/GIBBS/LIB
DATA  =/people/biostat5/hongzhe/GIBBS/DEC/CODE
LDFLAGS = -L$(GIBBS)
CFLAGS = -O  -I$(GIBBS)/Headers
LIB =   -lnghds -lrans -lm

CC      = gcc
OBJS    = gibbs.o  output.o
LKIN    = inputdata.o gene.o gen_alloc.o
SCOX1   = maxfun.o stdcox1.o

$(OBJS) : head.h
$(LKIN) : head.h
$(SCOX1): head.h

SOURCES = $(OBJS:.o=.c) $(LKIN:.o=.c) $(SCOX:.o=.c) $(SCOX1:.o=.c)

coxseg1 :coxseg.o $(OBJS) $(LKIN) $(SCOX1)
        $(CC) $(LDFLAGS) -o $@ coxseg.o $(OBJS) $(LKIN) $(SCOX1)$(LIB)

envidat2:coxseg1
        coxseg1 $(DATA)/envidat2 >envidat2.out
clean   : rm  -f *.o
-------------------------------------------------------------------------


Make Commands     Results
-------------------------------------------------------------------------
make clean        removes all executables and object code (.o files).
make coxseg1      creates executable file coxseg1.
make envidat2     run the program with Makefile for data set envidat2.
-------------------------------------------------------------------------
```

To run the program CoxGENE under UNIX, just type:

coxseg1 inputdata

In the shar file, we have a test data set *envidat2*, and the output file *envidat2.out*, and the *envidat2.est* file with parameter estimates.

Table 3: Input data for CoxGENE

line 1:	
column 1:	number of individuals
column 2:	always 0
column 3:	always 4
column 4:	2+number of covariates
column 5:	if 1, data entered as separate pedigrees,
	if 0, data entered as one large pedigrees.
line 2 - pedigrees data.	
column 1:	individual id;
column 2:	mother id;
column 3:	father id;
column 4:	sex (2=female; 1=male);
column 5:	age of onset or age at censoring;
column 6:	disease onset indicator (1=affected; 0=unaffected).
last 8 lines from the bottom.	
1st line after pedigree:	always 0;
2nd line after pedigree:	always 3 3;
3rd line after pedigree:	parameters for the Monte Carlo estimation:
column 1:	# scans/realization
column 2:	# EM iterations
column 3:	# realizations/EM iteration
column 4:	# scans of burn-in before initial sampling
column 5:	# final EM steps used to give final estimates
column 6:	# Newton-Raphson iteration
4th line after pedigree:	1 if sex is a covariate;
	0 if sex is not a covariate;
5th line after pedigree:	initial normal allele frequency;
6th line after pedigree:	initial risk ratio parameters;
column 1:	initial μ
column 2:	initial μ (same as column 1)
column 3:	initial β if there are covariates
7th line after pedigree:	output estimated baseline hazard function
8th line after pedigree:	parameter estimates.

2.4 Input Data Files

For CoxGENE program, pedigree data can be entered as one large pedigree, or many pedigrees. For data with many pedigrees, individual id being 1 is used to identify different pedigrees. This requires that for each family, the first individual has to have an individual id being 1 - the program will use this to identify different families. Table 2.4 explains line by line the input data file. An example is given after this table.

An Example of Input File for CoxGENE

```
----------------------------------
500 0 4 2 1
    1     0     0     2 37.7  1.0
    2     0     0     1 31.1  1.0
    3     0     0     2 44.0  1.0
    4     1     2     1 63.0  0.0
    5     1     2     2 48.8  1.0
    6     0     0     1 44.3  1.0
    7     3     4     2 39.5  0.0
    8     3     4     1 37.6  0.0
    9     5     6     2 24.8  1.0
   10     5     6     1 32.4  0.0
    1     0     0     2 29.5  1.0
    2     0     0     1 66.0  0.0
    3     0     0     2 31.5  1.0
    4     1     2     1 34.3  1.0
    5     1     2     2 25.6  1.0
    6     0     0     1 42.5  1.0
    7     3     4     2 30.6  1.0
    8     3     4     1 37.7  0.0
    9     5     6     2 26.1  1.0
   10     5     6     1 34.4  0.0
          .
          .
          .
    1     0     0     2 18.2  1.0
    2     0     0     1 61.4  0.0
    3     0     0     2 47.1  1.0
    4     1     2     1 26.0  1.0
    5     1     2     2 24.9  1.0
    6     0     0     1 64.0  0.0
    7     3     4     2 34.1  0.0
    8     3     4     1 36.5  0.0
    9     5     6     2 32.3  0.0
   10     5     6     1 31.9  0.0
0
3 3
10 500 500 200 10 100
1
0.70
1.000000 1.000000 1.0
envidat.lik
envidat2.est1
----------------------------------
```

2.5 Output Files

The output file include:

1. parameter values at each EM iteration. You can save this and plot it to check the convergence of EM algorithm,.
2. final estimates of the parameters (p, μ, β);
3. estimate of the cumulative hazard function. You can save this and get a plot of the estimated survival function.
4. estimated information matrix based on the partial likelihood. This matrix needs to be inverted to obtain the variance estimates.

Here is an example of the envidat2.est1 file:

```
****************************************
***Cox Model With Major Gene Effects****
***Written by Hongzhe Li            ****
****************************************

Date and time: Sun Mar 15 13:57:31 1998

1. Estimates of Beta  Assuming Independence
-------------------------------------------
Beta[ 0 ]= -0.811910

2. Initial Values for Parameters
--------------------------------
p, mu1=mu2,beta
0.700000 3.096472 3.096472 -0.730802

3. Start From Forward Gene-Dropping.
------------------------------------


4. Estimates for Every 1 Iterations
-----------------------------------
      p    mu(aA)=mu(AA)    beta's
(This is written in file - envidat2.hzd)

5. Estimates From Last EM Step
------------------------------
0.838750 3.668282 -1.162127

6. Estimates By Taking Averages
-------------------------------
0.829375 3.688112 -1.081347

7. Estimated Basline Hazards Function is written in
file envidat2.hzd

8. Final Estimates of Probability of Being a Carriers is
written in file envidat2.hzd

9. Estimated Information Matrix
-------------------------------
mu   beta's   p
```

```
13.863621 7.177823 -17.963450
7.177823 28.106287 -37.742679
-17.963450 -37.742679 562.916360
```

3 The CoxGENEENVI Package

3.1 Brief Description of CoxGENEENVI Routines

The programs distributed here need to call the Gibbs libraries maintained by Dr. Elizabeth Thompson at the University of Washington. See section 2.1 for details.

Table 4 gives a brief description of CoxGENEENVI Routines.

Table 4: Brief Description of CoxGENEENVI Routines

envi.c	main program when allele frequency is estimated
inputdata.c	input data
output.c	output data
gene.c	gene manipulation
gibbs.c	Gibbs sampling procedure
stdcox1.c	procedure for Cox regression model
maxfun.c	maximization functions
gamma.c	Some Gamma functions

3.2 Makefile for CoxGENEENVI

```
Makefile for CoxGENEENVI
------------------------------------------------------------------------
GIBBS = /people/biostat5/hongzhe/GIBBS/LIB
DATA  =/people/biostat5/hongzhe/GIBBS/ENVI/SIMUDATA
LDFLAGS = -L$(GIBBS)
CFLAGS  = -O  -I$(GIBBS)/Headers
LIB     = -lnghds -lrans -lm

CC      = gcc
OBJS    = gene.o gibbs.o output.o
LKIN    = inputdata.o  gamma.o gen_alloc.o
SCOX1   = stdcox1.o maxfun.o

$(OBJS) : head.h
$(LKIN) : head.h
$(SCOX1): head.h

SOURCES = $(OBJS:.o=.c) $(LKIN:.o=.c) $(SCOX:.o=.c) $(SCOX1:.o=.c)

envi1   :envi.o $(OBJS) $(LKIN) $(SCOX1)
        $(CC) $(LDFLAGS) -o $@ envi.o $(OBJS) $(LKIN) $(SCOX1) $(LIB)
envidat2        :envi1
        envi1 $(DATA)/envidat2 >envidat2.out
clean   :rm  -f *.o
------------------------------------------------------------------------


Make Commands     Results
-----------------------------------------------------------------
make clean        removes all executables and object code (.o files).
make envi1        creates executable file coxseg1.
make envidat2     run the program with Makefile for data set envidat2
-----------------------------------------------------------------
```

To run the program CoxGENEENVI under UNIX, just type:

envi inputdata

3.3 Input Data Files

The input data file for CoxGENEENVI is similar to that for the CoxGENE program. We list the difference here:
(1) line 1: no column 5;
(2) 6th line after pedigrees: column 1 is initial value for Gamma variance; column 2 and 3 are initial values for μ; column 3 is the initial value for β if there is covariate.

An example input data file is given in next page.

An Example of Input Data File for CoxGENEENVI

```
-------------------------------------------
500 0 4 2
    1     0     0     2 37.7  1.0
    2     0     0     1 31.1  1.0
    3     0     0     2 44.0  1.0
    4     1     2     1 63.0  0.0
    5     1     2     2 48.8  1.0
    6     0     0     1 44.3  1.0
    7     3     4     2 39.5  0.0
    8     3     4     1 37.6  0.0
    9     5     6     2 24.8  1.0
   10     5     6     1 32.4  0.0
    1     0     0     2 29.5  1.0
    2     0     0     1 66.0  0.0
    3     0     0     2 31.5  1.0
    4     1     2     1 34.3  1.0
    5     1     2     2 25.6  1.0
    6     0     0     1 42.5  1.0
    7     3     4     2 30.6  1.0
    8     3     4     1 37.7  0.0
    9     5     6     2 26.1  1.0
   10     5     6     1 34.4  0.0
                .
                .
                .
    1     0     0     2 18.2  1.0
    2     0     0     1 61.4  0.0
    3     0     0     2 47.1  1.0
    4     1     2     1 26.0  1.0
    5     1     2     2 24.9  1.0
    6     0     0     1 64.0  0.0
    7     3     4     2 34.1  0.0
    8     3     4     1 36.5  0.0
    9     5     6     2 32.3  0.0
   10     5     6     1 31.9  0.0
0
3 3
10 500 500 200 10 100
1
0.70
1.000000 1.000000 1.0
envidat.lik
envidat2.est1
-------------------------------------------
```

3.4 Output Files

Here is an example of the envidat2.est1 file outputed from the CoxGENEENVI package:

```
*********************************************************
*Cox Model With Major Gene and Familial Effects     ****
***         Written by Hongzhe Li                   ****
*********************************************************


Date and time: Wed Apr  1 11:04:02 1998

 1. A Summary of Data Set
 ------------------------
(a). Number of Affected: 195
(b). Average age of Onset is: 39.194872
(c). Number of families: 50

2. Parameters assuming independent
----------------------------------
Beta[ 0 ]= -0.811910

3. Initial Values for Parameters
--------------------------------
p, mu1=mu2, gamma, beta's
0.700000 3.096472 1.000000 -0.730802

4. Start From Forward Gene-Dropping
-----------------------------------

5. Estimates for Every 1 Iterations
p    mu(aA)=mu(AA))    theta    beta's
(This is written in file - envidat.lik)

6. Estimates From Last EM Step
------------------------------
0.550000 2.301535 0.954076 -1.015590

7. Estimates By Taking Averages
-------------------------------
0.572500 2.367988 0.968322 -0.977817

8. Estimated Baseline Hazards Function is written in file
envidat.lik


9. Predicted Probabilities of Being Carrier is
written in file envidat.lik


10. Estimated Information Matrix
--------------------------------
mu  beta's  theta  p
-33.134878 -23.029444 61.975401 -743.548649
-23.029444 -1.722423 16.885460 -235.188797
61.975401 16.885460 -36.156839 608.246988
```

-743.548649 -235.188797 608.246988 -6522.341244

4 The SibLINK Package

4.1 Brief description of the programs

SibLINK consists a collection of Splus functions for performig linkage analysis using the methods in Li and Zhong (2002). We assume the population disease rate is at least approximately known. Currently, the program can only handle the sib pair data. For a given test location (relative to a known map, user also need to obtain the probabilites of sib pair sharing 0,1, and 2 allels IBD at this location from running other programs such as MAPMAKER/SIBS programs. These probabilities are part of the input data.

4.2 Input Data Files

The function requires three input data files, *pheno, ibd, diseaserate*,

Table 5: *pheno*: Input phenotype data for SibLINK for sib pair data

column 1	family id;
column 2	individual ID;
column 3	disease status
column 4	age of onset or censoring
columns 5-	covariates

Table 6: *ibd*: Input IBD Sharind data for SibLINK for sib pair data. User need to run MAPMAKER/SIB to obtain these IBD sharing probabilities.

column 1	family id;
column 2	prob(IBD=0)
column 3	prob(IBD=1)
column 4	prob(IBD=2)

Finally, the user needs to input a rough estimate of the age-dependent disease rate data (*diseaserate*) from age 1 to maxage, where maxage is the max of age of onset or age at censoring.

4.3 R functions

USAGE:
SibLINK< −function(pehno,ibd,diseaserate,covlist=NULL,bb)

Input Data:

pheno	name of the Splus data set (see table 5).
ibd	IBD sharing probabilities (see table 6).
diseaserate	population disease rate for age 1,2,..,maxage (default 100).
covlist	index of covariates to be included in the analysis.
bb	initial values for the parameters (nud, nup, beta)

Output Data File The output includes a list of the following named components:
(1) converged - algorithm converged or not;
(2) objf - value of the loglikelihood function;
(3) nud - parameter estimate of ν_d;

(4) nup - parameter estimate of ν_p;
(5) beta - parameter estimate of β if covariates are included;
(6) Pvalue - p value from the likelihood ratio test of linkage.

5 The PCRCox R Package

5.1 Brief description of the programs

This set of R functions is written to generate the partial Cox regression components as defined in Li and Gui (2004) for high-dimensional gene expression data and censored time to response outcomes. These components can then be used as predictors in the regular Cox regression analysis. The number of the components used in the model are usually the first several PCR components determined by univariate analysis. See Section 9 for an example using simulated data set.

5.2 R functions

USAGE:
PCRCoxf< − function(x, time, surv.st, k)
Description: This function automatically centers and standardizes predictors, and computes the k PCR scores.

Input:
x: n*m predictor matrix, with m predictors, n samples. Note that x can be original gene expression variables, or the principal components derived from the original variables.
time: survival time for n observations.
surv.st: censoring status for n observations (1=event, 0=censoring.
k: number of output PCR components.

Output:
program outputs k PCR components in "comp" for each individual.
each PCR's corresponding coefficients outputs are in "coef".

5.3 An Example

For an example of using this function on simulated data set, see Section 9.

6 The TGDCoxPackage

6.1 Brief description of the programs

This set of R functions is written to implement the threshold gradient descent procedure for the Cox regression analysis in the high-dimensional and low-sample size setting. The methods are described in Gui and Li (PSB 2005) using the idea by Friedman and Popescu (2004). For a given threshold value τ, we use cross-validated partial likelihood to choose the step. See Section 9 for an example using simulated data set.

6.2 R Functions

There are a total of 4 functions, *gdcvpl* calls *gd*, *gd* calls *lik1*, *findbeta* uses the output of *gd* and *gdcvpl* as input. To use these functions, first use *gdcvpl* to get the optimal step value "stepno" for given threshold. Then run *gd* and use the output "gra" and "stepno" from *gdcvpl* to calculate the estimates of β.

USAGE:
gd <− function(x, pred, predtime, predsurv, time, surv.st, thres, epi, maxstep)
Description: For a given threshold value *thres*, for each step (1 to maxstep), this function computes the gradient for training score and scores for testing data set. Function "gdcvpl" calls this function for cross-validation analysis to determine which step to stop the iterations.

Input:
x: n*m predictor matrix, with m predictors and n samples.
time: survival time for n observations.
surv.st: censoring status for n observations (1=event; 0=censored).
x, time, surv.st together are training dataset
pred, predtime, predsurv: together are corresponding testing dataset
thres: threshold value between 0 and 1
epi: maximum factor (step size) scaling gradient for incrementing selected coefficients at each step
maxstep: maximum number of threshold gradient descent iterations.

Output:
gra: n*maxstep, is the gradient for the training score (x%*%beta) in each step.
pred: n1*maxstep, is the score(pred%*%beta) for testing dataset in each steps.
lik: testing data's partial likelihood in each steps.
cvpl: will be used to calculate the cross-validated partial likelihood in "gdcvpl".

USAGE:
gdcvpl< − function(x, time, surv.st, thres, epi, maxstep, nfold)
Description: This function calculates the cross-validated partial likelihood score for a given "thres" and for each step from 1 to maxstep. This CVPl scores are used to determine the step to stop the iterations. It calls function *gd*.

Input:
x: n*m predictor matrix, with m predictors.
time: survival time for n observations.
surv.st: censoring status for n observations.
thres: threshold value between 0 and 1
epi: maximum factor (step size) scaling gradient for incrementing selected coefficients at each step
maxstep: maximum number of threshold gradient descent iterations
nfold: number of cross-validation iterations.

Output:
program outputs the cross-validated partial likelihood for given epi and thres for each iteraction from 1:maxstep. The output includes: epi=episilon used,
cvpl=cvplscore from step 1 to maxstep,

op= step # where the max CVPL is achieved.

USAGE:
findbeta <− function(x, gra, thres, epi, stepno)
Description: After CVPL determines the "stepno", this function calculate the β values.

Input:
x: n*m predictor matrix, with m predictors.
gra: output from "gd", gradient for x%*%beta in each steps
thres: threshold value between 0 and 1
epi: maximum factor (step size) scaling gradient for incrementing selected coefficients at each step
note that thres and epi should be same as what is in "gd".

Output:
the estimated β in step "stepno".

USAGE:
lik1 <− function(score, time, surv.st)
Description: utility function to return the partial likelihood for Cox model.

6.3 Example

See Section 9 for an example of using this function.

7 The LarsCox Package

7.1 Brief description of the programs

This set of Splus functions is written to fit the Lasso (Tibshrani 1997) for the Cox regression by using the LARS procedure (Efron et al 2004) as described in Gui and Li (2005). We slightly modified the Splus function provided by Efron et al by not standardizing the data matrix. User needs to download the LARS functions written by Hastie at el.

$$http://www-stat.stanford.edu/\sim hastie/Papers/LARS/larfuncs.S$$

in order to run the program and makes the following slight changes to the *lars* function,

```
### Center x and y, and scale x, and save the means and sds
  meanx <- drop(one %*% x)/n
  meanx<-rep(0,m)  ###added by Jiang Gui
  x <- scale(x, meanx, FALSE)   # centers x
  normx <- sqrt(drop(one %*% (x^2)))
  normx<-rep(1,m)    ###added by Jiang Gui
```

We use cross-validated partial likelihood to choose the tuning parameter. Currently, the programs only run in Splus since it uses the Splus function *chol* for Cholesky decomposition of a singular matrix, which is not supported by R. We are now writing a R function for this. See Section 9 for an example using simulated data set.

7.2 Splus functions

There are a total of four functions, *L1cvpl* calls *surv4*, *surv4* calls *surv1*, and *surv1* calls *lars*, where *lars* is slight modification of the "lars" function provided by Efron and Hastie (simply not to standardize the X matrix and Y). *lars* can be downloaded from the website of Hastie at Stanford University. To use these functions, first use *L1cvpl* to perform cross-validation for each L1 constraint (tuning parameter). Pick a tuning parameter (usually plot the CVPL scores vs the tuning parameter) and use *surv4* to obtain the corresponding parameter estimates of β.

USAGE:
surv1<- function(x, time, surv.st, beta, ttt)
Description: This function calls LARS to solve the IRWL step 3 for a given iteration step *ttt*.

Input:
x: n*m predictor matrix, with m predictors.
time: survival time for n observations.
surv.st: censoring status for n observations.
beta: initial value for beta.

Output:
the estimate of β at step ttt.

USAGE:
surv4 <- function(x, time, surv.st, ss)
Description: For a given L_1 constraint *ss*, this function estimates the βs. This function calls *surv*1 for performing IRWL using *lars*.

Input:
x: n*m predictor matrix, with m predictors.
time: survival time for n observations.
surv.st: censoring status for n observations.
ss: the L1 constraint for β. ss could be a vector in acsending order.

Output:
the estimate of β at each L1 constraint in ss.

USAGE:
L1cvpl <- function(x, time, surv.st, ss, nfold)
Description: This function calculate the CVPL score for each *ss*, which can be used to determine the tuning parameter *ss*. This function calls *surv*4.

Input:
x: n*m predictor matrix, with m predictors.
time: survival time for n observations.
surv.st: censoring status for n observations.
ss: the L1 constraint for β. ss could be a vector in acsending order.
nfold: number of cross-validation iterations.

Output:
program output the cross validated partial likelihood at given ss. User call plot L1cvpl vs ss to determine the tuning parameter ss.

USAGE:
lik<- function(x, time, surv.st, beta)
Description: Utility function to return the partial likelihood for Cox model

7.3 Example

See Section 9 for an example using simulated data set.

8 REFERENCES

Friedman JH and Popescu BE (2004): Gradient Directed Regularization for Linear Regression and Classification. *Technical Report, Statistics Department, Stanford University.*

Gui J and Li H (2005): Penalized Cox Regression Analysis in the High-Dimensional and Low-sample Size Settings, with Applications to Microarray Gene Expression Data. *Bioinformatics*, 21: 3001-3008.

Gui J, Li H (2005): Threshold Gradient Descent Method for Censored Data Regression, with Applications in Pharmacogenomics. *Pacific Symposium on Biocomputing*, 10: 272-283.

Li H, Gui J (2004): Partial Cox Regression Analysis for High-Dimensional Microarray Gene Expression Data. *Bioinformatics*, 20:i208-i215.

Li H, Thompson EA (1997). Semiparametric estimation of major gene and random familial effects for age of onset. *Biometrics* **53**, 282-293.

Li H, Thompson EA, Wijsman EA (1998). Semiparametric estimation of major gene effects for age of onset. *Genetic Epidemiology*, 15:279-298.

Li H, Zhong X (2002): *Multivariate survival models induced by genetic frailties, with application to linkage analysis. Biostatistics*, 3:57-75.

Tibshirani, R. (1997):The Lasso method for variable selection in the Cox model. *Statistics in Medicine* 16,385-395.

Efron B, Johnston I, Hastie T and Tibshirani R (2004): Least angle regression. *Annals of Statistics*, 32: 407-499.

9 APPENDIX 1: Examples of using the R/Splus Functions (PCRCox, TGDCox, and LarsCox) using Simulated Data sets

10 APPENDIX 2- Splus/R codes for RCRCox, TGDCox and LarsCox Packages

11 APPENDIX 3- SAM-GA Website for Downloading the Programs